ID Generation in Mobile Environments

Paulo Jesus, Carlos Baquero, and Paulo Almeida

Minho University, Campus de Gualtar, 4710-057 Braga, Portugal,
paulo.jesus@ieee.org,
cbm@di.uminho.pt,
psa@di.uminho.pt

Abstract. This work is focused in the ID generation problem in mobile environments. We discuss the suitability of traditional mechanisms and techniques to generate IDs in mobile environments. Based on the "Birthday Problem", we deduced some formulas to evaluate the ID trust that is directly related to the number of entities in the system. The estimation of the system size revels to be the main problem of our approach. To deal with it, we develop a recursive schema that needs to be evaluated. Alternatively, we also design an aggregation algorithm to estimate the system size, which results are currently been analyzed.

1 Introduction

In the last years, we assisted to an increasing use of mobile device resulting in an increased importance of mobile systems in our quotidian life (as mobile phones already do). The characteristics of mobile computing systems bring new challenges and problems, departing from classical distributed systems assumptions.

Unique identification is widely used in all kind of scenarios. In the computational world, unique identification is used from the hardware level (to identify products and components) to the software level (to identify several elements like information data and communicating entities). Considering a dynamic mobile environment were new devices can appear and disappear at any time, it is often necessary to generate a system wide unique identification for a new mobile entity, in order to allow its interaction within the system. This study exposes the problem of ID generation in mobile environments and presents an approach that aims to help solve this problem.

2 Unique Identifiers

Traditional computing environments and applications use IDs for several proposes and in different levels of software development. For example: at the network level, communication in the IPv4 protocol can lead to the generation of IDs in order to identify Internet datagrams during data transmission, consequence of the fragmentation process (see: RFC 791 [1]); it is usual to use a unique key identification in databases; Operating systems like Microsoft Windows use IDs to reference data and object in the registry file; in FEW [2] (Files EveryWhere), a distributed system that manages and synchronizes file replicas in a mobile environment, there is the need to uniquely identify each unit of replication.

The need to uniquely identify information in a non-centralized way lead the OSF (Open Software Foundation) to specify a class of UUID [3] (Universally Unique Identifiers) that become a standard. There are ongoing efforts to document the UUID standard in separate, through the ISO (International Organization for Standardization) [4] and the IETF (Internet Engineering Task Force) [5].

Traditional schemas to generate unique identification are based in known mechanisms (time-based, name-based and random-based) and techniques (non-centralized, centralized and distributed). The UUID standard specifies five different versions[1] of ID generation that are based in three mechanisms: Time-Based; Name-Based and Random-Based.

A non-centralized technique like UUID appears to be appropriate for use in a mobile environment, allowing autonomous ID generation which is compatible with disconnected operation. However, the UUID standard presents several weaknesses: the use of fixed length identifiers can be argued to generate identifiers that are too large to use in devices with few memory resources like sensors; conversely, they could be found too small to use in systems like “dust networks” with a huge quantity of devices. The solution to these shortcomings could be approached by resorting to variable length IDs or calibration of ID length after probing the deployment environment.

On the opposite extreme, we find centralized ID generation techniques where uniqueness of identifiers is guaranteed. For example, some schemas of license key registration, like the one used by Google Web APIs [6] to provide a limited and controlled access to web services. This kind of approach does not adequately work in a mobile environment where the communication bandwidth and the channel conditions are variable, and is necessary to consider working in disconnected mode. A centralized ID generation architecture requires a mandatory connection with a centralized entity that generates the IDs. Due to the nature of mobile environments this constraint cannot often be supported.

In distributed ID generation technique like the one used in Bayou [7] and Panasync [8], new IDs are created recursively from any available server. These mechanisms of ID creation only requires one connection to any given available server, which allows a more flexible way to create IDs while preserving uniqueness across the whole system. These approaches are more appropriate for mobile environments where it can be tolerated that in a specific moment (ID creation) it is required a single connection with any entity of the system.

As we can see, several ID generation techniques are used in the computational world but few are suitable for mobile environments.

[1] Version 1: time-based; version 2: DCE security; version 3: name-based using MD5 hashing; version 4: random-based; version 5: name-based using SHA-1 hashing.

3 Collision on Random IDs

Generally the intention of using IDs is to uniquely identify a specific entity. ID collision occurs when two different entities create the same ID to refer distinct targets which should be differentiated by their unique identification. This occurrence breaks the objective of using unique identifiers. In a system where the IDs generation process does not guaranty that no collisions happen, the consequences can be unpredictable and compromise the correct execution of the system.

Centralized and recursive ID generation techniques are generally "collision secure", giving full trust that no ID collision will take place. Techniques that are not "collision secure" usually use methods that estimate a low probability of ID collision, considering that the probability of generating the same ID is so small that it can be ignored. This is the case of the UUID standard that is optimistic with respect to the possibility of generating the same UUID.

The study of collision probabilities can be analyzed with precision. Based on the "Birthday Problem" (introduced by Von Mise in 1932 as referred by Anirban DasGupta in [9]) a set of formulas can be deduced allowing: The calculation of the ID trust (probability of no ID collision) for a given number of elements and a specified number of possible unique identifiers (uniformly distributed); The estimation of the number of elements that can exist for a particular universe of possible unique identifiers and a given minimum ID trust probability; Calculating the minimum length l (in bits) that a binary unique identifier should have, to guaranty a given probability P of ID trust, considering the existence of n elements.

This last case is shown in the formula below:

$$l \approx \log_2 \left(\frac{n(n-1)}{2.\ln(1/P)} \right) \tag{1}$$

4 Approach

We are working in a dynamic ID generation approach that maintains a fixed ID trust as the mobile system grows. To achieve this goal the length of new created IDs will vary according to the number of entities in the system, resulting in the coexistence of IDs with different sizes. Formula (1) can be used to calculate the ID length according to the population size and a given ID trust, assuming the use of a function that generates IDs with a uniform distribution.

Since the ID length can be calculated and the ID trust is fixed, the remaining value that needs to be addressed is the population size. The knowledge of the number of entities in the system represents the main problem which needs to be solved. Existing techniques and algorithms that address this problem are essentially based in aggregation protocols [10] [11], but are not adapted to mobile environments reality. To deal with this issue we design a new recursive schema, that keep track of known IDs in a compressed way through the use of Bloom Filters [12]. This approach need to be accurately evaluated. Alternatively, new aggregation algorithms to estimate the system size are been considered.

The main objective of our study is to cover the generation of system wide and permanent unique IDs in mobile environments, considering all possible communication and interaction patterns. An entity may eventually need an initial identification to establish a local communication with a neighbor entity in range. For this propose, an initially temporary random ID with a conservative length can be used, but should not thereafter pollute the ID space. After the initial communication, eventually performed with the temporary ID, and once the network size is estimated, a more permanent random ID can be generated with an adequate length that guarantees system wide uniqueness with high probability.

Our current work is focused in algorithms for system size estimation that can operate in an almost anonymous setting and bootstrap the subsequent ID generation.

References

1. Postel, J.: RFC 791: Internet Protocol (1981) Obsoletes RFC 760. Status: STANDARD.
2. Preguiça, N., Baquero, C., Martins, J.L., Shapiro, M., Almeida, P.S., Domingos, H., Fonte, V., Duarte, S.: FEW: File Management for Portable Devices. Proceedings of The International Workshop on Software Support for Portable Storage (2005)
3. 11578:1996, I.S.I.: (Information technology open systems interconnection Remote Procedure Call (RPC))
4. 9834-8, I.S.I.: Information technology - open systems interconnection: Procedures for the operation of OSI registration authorities: Generation and registration of Universally Unique Identifiers (UUIDs) and their use as ASN.1 object identifier components. In: ITU-T Recommendation X.667, Prepublished version (2004)
5. IETF Network Working Group: A UUID URN Namespace, IETF Network Working Group, Internet-Draft (2004)
6. web APIs, G.: Internet site: http://www.google.com/apis/index.html ((last access: 18 March 2006))
7. Petersen, K., Spreitzer, M.J., Terry, D.B., Theimer, M.M., Demers, A.J.: Flexible update propagation for weakly consistent replication. In Sixteen ACM Symposium on Operating Systems Principles (1997)
8. Almeida, P.S., Baquero, C., Fonte, V.: Panasync: dependency tracking among file copies. In: EW 9: Proceedings of the 9th workshop on ACM SIGOPS European workshop, New York, NY, USA, ACM Press (2000) 7–12
9. DasGupta, A.: The matching, birthday and the strong birthday problem: a contemporary review. Journal of Statistical Planning and Inference **130**(1-2) (2005) 377–389
10. Jelasity, M., Montresor, A., Babaoglu, O.: Gossip-based aggregation in large dynamic networks. ACM Trans. Comput. Syst. **23**(3) (2005) 219–252
11. Horowitz, K., Malkhi, D.: Estimating network size from local information. Inf. Process. Lett. **88**(5) (2003) 237–243
12. Bloom, B.H.: Space/time trade-offs in hash coding with allowable errors. Commun. ACM **13**(7) (1970) 422–426